

Abstract of the Disclosure

1  
2  
3 The invention provides an improved method and apparatus for creating a  
4 snapshot of a file system. In a first aspect of the invention, a "copy-on-write" mechanism  
5 is used. An effective snapshot mechanism must be efficient both in its use of storage  
6 space and in the time needed to create it because file systems are often large. The  
7 snapshot uses the same blocks as the active file system until the active file system is  
8 modified. Whenever a modification occurs, the modified data is copied to a new block  
9 and the old data is saved (henceforth called "copy-on-write"). In this way, the snapshot  
10 only uses space where it differs from the active file system, and the amount of work  
11 required to create the snapshot is small. In a second aspect of the invention, a record of  
12 which blocks are being used by the snapshot is included in the snapshot itself, allowing  
13 effectively instantaneous snapshot creation and deletion. In a third aspect of the  
14 invention, the state of the active file system is described by a set of metafiles; in  
15 particular, a bitmap (henceforth the "active map") describes which blocks are free and  
16 which are in use. The inode file describes which blocks are used by each file, including  
17 the metafiles. The inode file itself is described by a special root inode, also known as the  
18 "fsinfo block". The system begins creating a new snapshot by making a copy of the root

1 inode. This copy of the root inode becomes the root of the snapshot. The root inode  
2 captures all required states for creating the snapshot such as the location of all files and  
3 directories in the file system, it. During subsequent updates of the active file system, the  
4 system consults the bitmap included in the snapshot (the "snapmap") to determine  
5 whether a block is free for reuse or belongs to the snapshot. This mechanism allows the  
6 active file system to keep track of which blocks each snapshot uses without recording any  
7 additional bookkeeping information in the file system. In a fourth aspect of the  
8 invention, a snapshot can also be deleted instantaneously simply by discarding its root  
9 inode. Further bookkeeping is not required, because the snapshot includes it's own  
10 description. In a fifth aspect of the invention, the performance overhead associated with  
11 the search for free blocks is reduced by the inclusion of a summary file. The summary  
12 file identifies blocks that are used by at least one snapshot; it is the logical OR of all the  
13 snapmap files. The write allocation code decides whether a block is free by examining  
14 the active map and the summary file. The active map indicates whether the block is  
15 currently in use in the active file system. The summary file indicates whether the block is  
16 used by any snapshot. In a sixth aspect of the invention, the summary file is updated in  
17 the background after the creation or deletion of a snapshot. This occurs concurrently  
18 with other file system operations. Two bits are stored in the file system "fsinfo block"

1 for each snapshot. These two bits indicate whether the summary file needs to be updated  
2 using the snapshot's snapmap information as a consequence of its creation or deletion.  
3 When a block is freed in the active file system, the corresponding block of the summary  
4 file is updated with the snapmap from the most recently created snapshot, if this has not  
5 already been done. An in-core bit map records the completed updates to avoid repeating  
6 them unnecessarily. This ensures that the combination of the active bitmap and the  
7 summary file will consistently identify all blocks that are currently in use. Additionally,  
8 the summary file is updated to reflect the effect of any recent snapshot deletions when  
9 freeing a block in the active file system. This allows reuse of blocks that are now entirely  
10 free. After updating the summary file following a snapshot creation or deletion, the  
11 corresponding bit in the fsinfo block is adjusted. In a seventh aspect of the invention, the  
12 algorithm for deleting a snapshot involves examining the snapmaps of the deleted  
13 snapshot and the snapmaps of the next oldest and next youngest snapshot. A block that  
14 was used by the deleted snapshot but is not used by its neighbors can be marked free in  
15 the summary file, as no remaining snapshot is using it. However, these freed blocks  
16 cannot be reused immediately, as the snapmap of the deleted snapshot must be preserved  
17 until summary updating is complete. During a snapdelete, free blocks are found by using  
18 the logical OR of the active bitmap, the summary file, and the snapmaps of all snapshots

1 for which post-deletion updating is in progress. In other words, the snapmap of the  
2 deleted snapshot protects the snapshot from reuse until it is no longer needed for  
3 updating. In the preferred embodiment, the invention is operative on WAFL file system.  
4 However, it is still possible for the invention to be applied to any computer data storage  
5 system such as a database system or a store and forward system such as cache or RAM if  
6 the data is kept for a limited period of time.

008T80" F9024960